Object Recognition

Goal: Introduce central issues of object recognition, basic techniques, and emerging directions.

Outline:

1. What is object recognition and why is it challenging?
2. Historical perspective
3. Basic view-based classifiers and common problems
4. Multi-Class recognition
5. Scene and Geometric Context
6. Parsing and Segmentation

Optional Readings:

- J. Mundy. ”Object recognition in the geometric era: A retrospective”
- S. Dickinson. ”The evolution of object categorization and the challenge of image abstraction”

Matlab Tutorials and Demo Code:

- SIFT
- Boosted classifier (A. Torralba):

  `http://people.csail.mit.edu/torralba/shortCourseRLOC/boosting/boosting.html`

Background

What is object recognition?

- validation
- detection
- instance recognition (identification)
- category recognition
- scene/context recognition
- activity recognition

Challenges

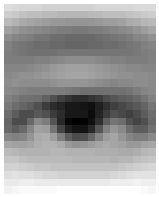
- variation in view point and lighting
- variation in shape, pose and appearance
- clutter and occlusion
- function versus morphology

Historical Perspective

- Blocks world
- 3D shape and part decomposition
- Perceptual organization
- Appearance-based models
- Context (3D and 2D) and Parsing

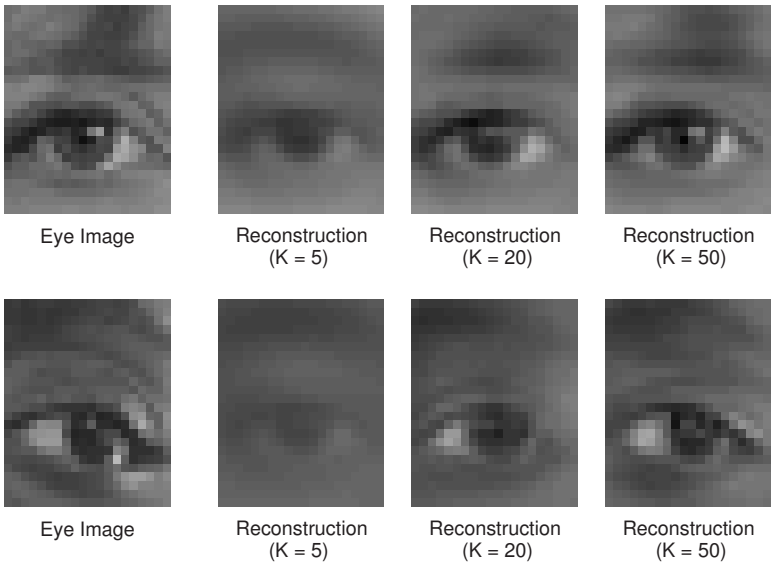
Subspace Models for Detection

Mean Eye:



Basis Images $(1-6, \text{ and } 10:5:35)$**:**

Reconstructions (for $K = 5, 20, 50$)**:**



Subspace Models for Detection

Generative model, $\mathcal{M}$, for random eye images:

$$\vec{\mathbf{I}} \;=\; \vec{\mathbf{m}} + \left(\sum_{k=1}^{K} a_k \vec{\mathbf{b}}_k \right) \;+\; \vec{\mathbf{e}}$$

where $\vec{\mathbf{m}}$ is the mean eye image, $a_k \sim \mathcal{N}(0, \sigma_k^2)$, σ_k^2 is the sample variance associated with the k^{th} principal direction in the training data, and $\vec{\mathbf{e}} \sim \mathcal{N}(0, \sigma_e^2 \mathbf{I}_{N^2})$ where $\sigma_e^2 = \frac{1}{N^2} \sum_{k=K+1}^{N^2} \sigma_k^2$ is the per pixel out-of-subspace variance.

Random Eye Images:

Random draws from generative model (with K = 5, 10, 20, 50, 100, 200)

So the likelihood of an image under this model of eyes is

$$p(\vec{\mathbf{I}} \,|\, \mathcal{M}) \;=\; \left(\prod_{k=1}^{K} p(a_k | \mathcal{M}) \right) \; p(\vec{\mathbf{e}} \,|\, \mathcal{M})$$

where

$$p(a_k | \mathcal{M}) \;=\; \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{a_k^2}{2\sigma_k^2}} \quad , \qquad p(\vec{\mathbf{e}} \,|\, \mathcal{M}) = \prod_{j=1}^{N^2} \frac{1}{\sqrt{2\pi}\sigma_e} e^{-\frac{e_j^2}{2\sigma_e^2}} \quad .$$

Eye Detection

The log likelihood of the model is given by

$$\begin{aligned} L(\mathcal{M}) \equiv \log p(\vec{\mathbf{I}} \,|\, \mathcal{M}) &= \left(\sum_{k=1}^{K} \log p(a_k | \mathcal{M}) \right) + \log p(\vec{\mathbf{e}} \,|\, \mathcal{M}) \\ &= \left(\sum_{k=1}^{K} \frac{-a_k^2}{2\sigma_k^2} \right) + \left(\sum_{j=1}^{N^2} \frac{-e_j^2}{2\sigma_e^2} \right) + const \\ &\equiv S_{in}(\vec{\mathbf{a}}) + S_{out}(\vec{\mathbf{e}}) + const \end{aligned}$$

Detector:

1. Given an image $\vec{\mathbf{I}}$
2. Compute the subspace coefficients $\vec{\mathbf{a}} = \mathbf{B}^T(\vec{\mathbf{I}} - \vec{\mathbf{m}})$
3. Compute residual $\vec{\mathbf{e}} = \vec{\mathbf{I}} - \vec{\mathbf{m}} - \mathbf{B}\vec{\mathbf{a}}$
4. For $S(\vec{\mathbf{a}}, \vec{\mathbf{e}}) = S_{in}(\vec{\mathbf{a}}) + S_{out}(\vec{\mathbf{e}})$, and a given threshold τ, the image patch is classified as an eye when

$$S(\vec{\mathbf{a}}, \vec{\mathbf{e}}) > \tau \ .$$

Eye Detection

Terminology:

- true positive = hit
- true negative = correct rejection
- false positive = false alarm (type I error)
- false negative = miss (type II error)

	classified positives	classified negatives	
true examples	true positives, T_{pos}	false negatives, F_{neg}	$N_{pos} = T_{pos} + F_{neg}$
false examples	false positives, F_{pos}	true negatives, T_{neg}	$N_{neg} = F_{pos} + T_{neg}$
	C_{pos}	C_{neg}	N

Definitions:

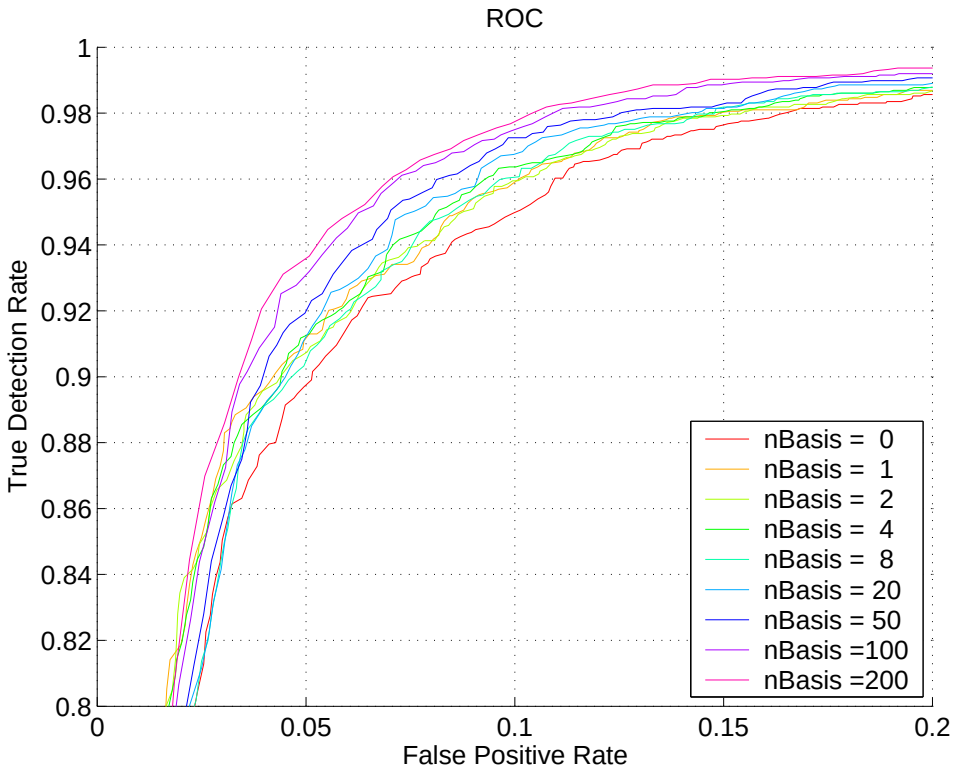
- true positive (hit) rate: $\rho_{tp} = T_{pos}/N_{pos}$ (sensitivity)
  (i.e., what fraction of the true eyes do we find?)
- true negative (reject) rate: $\rho_{tn} = T_{neg}/N_{neg}$ (specificity)
- false positive rate: $\rho_{fp} = F_{pos}/N_{neg} = 1 - \rho_{tn}$ (1 - specificity)

- precision: T_{pos}/C_{pos}
  (i.e., what fraction of positive response are correct hits? ...
  i.e., how noisy is the detector?)
- recall: $\rho_{tp} = T_{pos}/N_{pos}$
  (i.e., what fraction of the true eyes do we actually find?)

Eye Detection

ROC Curves:

- true detection rate (sensitivity) vs false positive rate (1-specificity)
- trade-off (as a function of decision threshold τ) between sensitivity (hit rate) and specificity (responding only to positive cases)



Here the eye images in the test set were different from the those in the training set. Non-eyes were drawn at random from images.

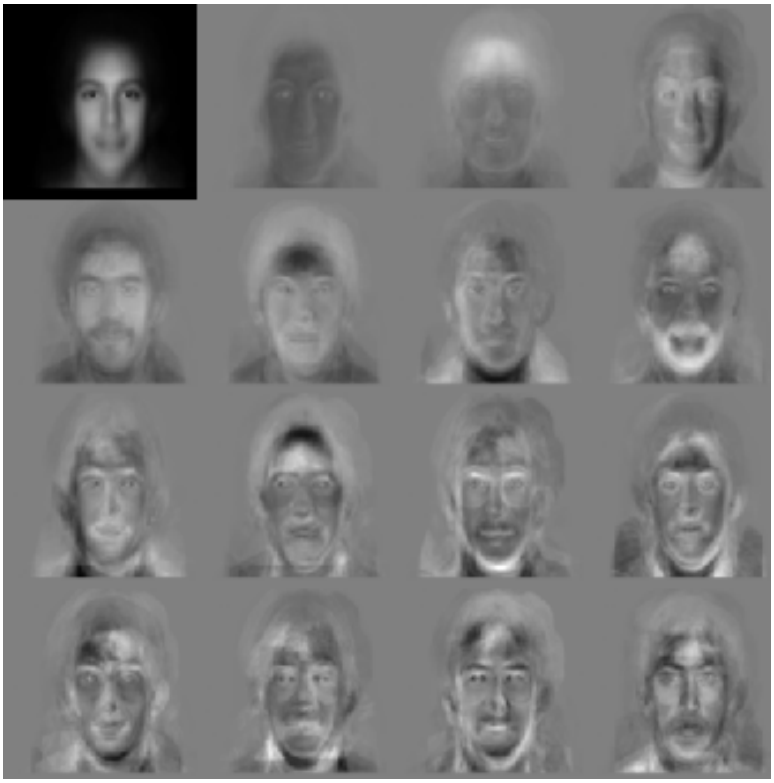
Precision-Recall Curves:

- precision vs true detection rate (sensitivity)
- better whan ROC when the $N_{neg} \gg N_{pos}$, so even a low false positive rate can yield many more false alarms than hits.
- that's why precision divides true hits by total number of hits rather than total number of positives.

Face Detection

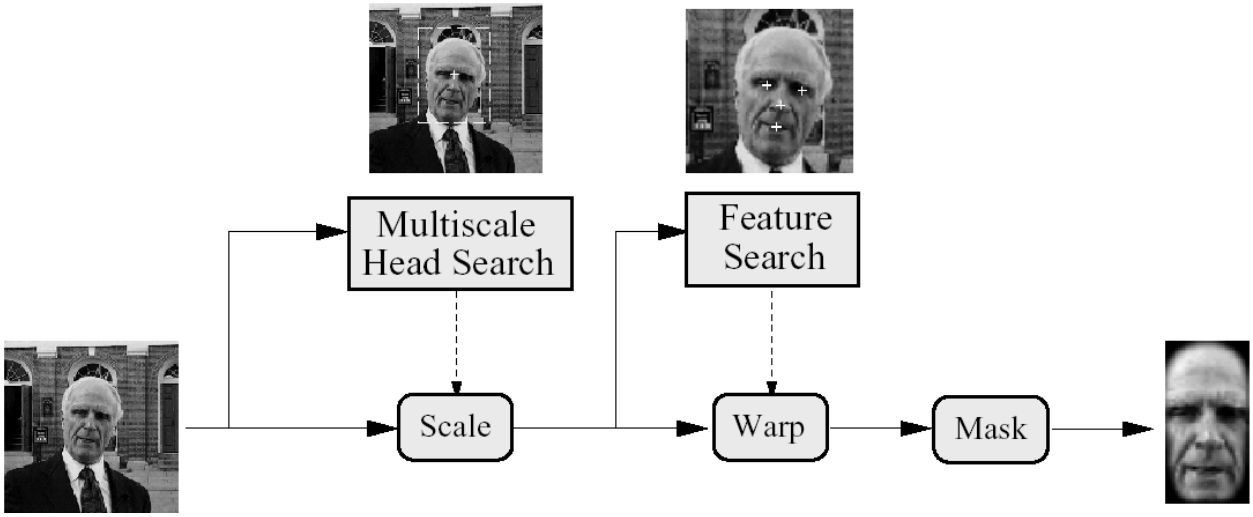
The wide-spread use of PCA for object recognition began with the work Turk and Pentland (1991) for face detection and recognition.

Shown below is the model learned from a collection of frontal faces, normalized for contrast, scale, and orientation, with the backgrounds removed prior to PCA.



Here are the mean image (upper-left) and the first 15 eigen-images. The first three show strong variations caused by illumination. The next few appear to correspond to the occurrence of certain features (hair, hairline, beard, clothing, etc).

Face Detection/Recognition



Moghaddam, Jebara and Pentland (2000): Subspace methods are used for head detection and then feature detection to normalize (warp) the facial region of the image.

Recognition: Are these two images (test and target) the same?

Approach 1: *Single Image Subspace Recognition:*

Project test and target faces onto the face subspace, and look at distance within the subspace.

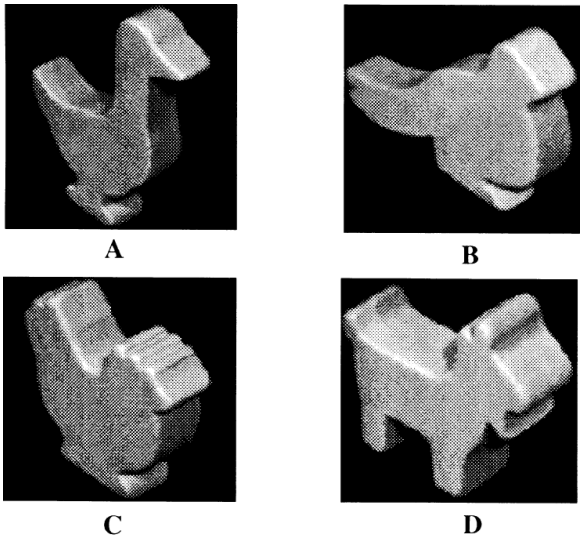
Approach 2: *Intra/Extra-Personal Subspace Recognition:*

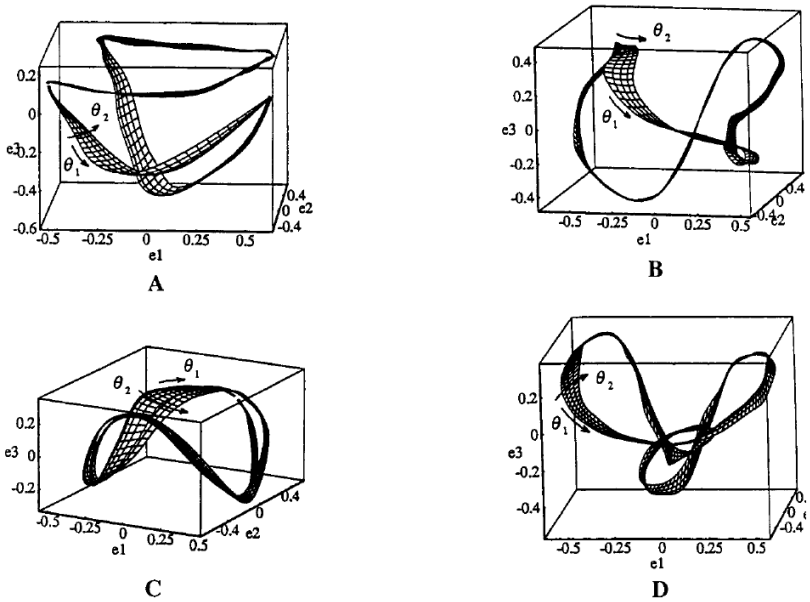
- An intra-personal subspace is learned from difference images of the same persion under variation in lighting and expression.
- The extra-personal subspace learned from difference between images of different people under similar conditions.

Object Recognition

Murase and Nayar (1995)

- images of multiple objects, taken from different positions on the viewsphere
- each object occupies a manifold in the subspace (as a function of position on the viewsphere)
- recognition: nearest neighbour assuming dense sampling of object pose variations in the training set.





Gaussian Class-Conditional Models

In general, suppose we have Gaussian class-conditional models for the image feature vector $\vec{\mathbf{d}}$. For models M_k, we model the observation densities $p(\vec{\mathbf{d}} \mid M_k)$ with Gaussians.

E.g., for two classes, M_1 and M_2, let the prior probabilities be $p(M_1)$ and $p(M_2) = 1 - p(M_1)$. The observation densities are Gaussian with means $\vec{\mu}_k$ and covariances C_k (for $k = 1, 2$).

Then, the posterior probability for model M_k given the data $\vec{\mathbf{d}}$

$$p(M_k \mid \vec{\mathbf{d}}) \;=\; \frac{p(M_k)\, G(\vec{\mathbf{d}};\, \vec{\mu}_k,\, C_k)}{p(\vec{\mathbf{d}})}.$$

The log odds $\delta(\vec{\mathbf{d}}\,)$ for model M_1 over M_2 is defined to be

$$\begin{aligned}
\delta(\vec{d}) \;&\equiv\; \log\left[\frac{p(M_1 \mid \vec{\mathbf{d}}\,)}{p(M_2 \mid \vec{\mathbf{d}}\,)}\right] \\
&=\; \log\left[\frac{p(M_1)\;|C_2|^{1/2}}{p(M_2)\;|C_1|^{1/2}}\right] \;+ \\
&\qquad \frac{1}{2}\left[(\vec{\mathbf{d}} - \vec{\mu}_2)^T C_2^{-1}(\vec{\mathbf{d}} - \vec{\mu}_2) - (\vec{\mathbf{d}} - \vec{\mu}_1)^T C_1^{-1}(\vec{\mathbf{d}} - \vec{\mu}_1)\right] \quad \textbf{(1)}
\end{aligned}$$

Thresholding the log odds at zero yields the decision boundary.

The decision boundary is a quadratic surface in $\vec{d}$ space (a quadratic discriminant). When both classes have the same covariance, i.e., $C_1 = C_2$, the quadratic terms in (1) cancel and the decision boundary becomes a hyperplane.

Logistic Regression

Let's return to the posterior class probability:

$$P(M_1 \,|\, \vec{\mathbf{d}}) = \frac{p(\vec{\mathbf{d}} \,|\, M_1)P(M_1)}{p(\vec{\mathbf{d}} \,|\, M_1)P(M_1) + p(\vec{\mathbf{d}}|M_2)P(M_2)} \,. \tag{2}$$

Dividing the numerator and denominator by $p(\vec{\mathbf{d}}|M_1)P(M_1)$ gives:

$$P(M_1|\vec{\mathbf{d}}) \;=\; \frac{1}{1+e^{-a(\vec{\mathbf{d}})}} \quad , \quad a(\vec{\mathbf{d}}) = \ln \frac{p(\vec{\mathbf{d}}|M_1)P(M_1)}{p(\vec{\mathbf{d}}|M_2)P(M_2)} \,. \tag{3}$$

The posterior probability of M_1 grows as a grows, and when $a = 0$, the posterior is $P(M_1|\vec{\mathbf{d}}) = \frac{1}{2}$; i.e., $a(\vec{\mathbf{d}}) = 0$ is the decision boundary.

Let's assume a linear decision boundary (independent of any assumptions about or having to fit the observation densities). That is, let

$$a(\vec{\mathbf{d}}) \;=\; \vec{\mathbf{w}}^T \vec{\mathbf{d}} + b \tag{4}$$

To learn a classifier, given IID training exemplars, $\{\vec{\mathbf{d}}_j, y_j\}$, where $y_j = \{1, 2\}$, we minimize the negative log likliehood:

$$\begin{aligned} \log p(\{\vec{\mathbf{d}}_j, y_j\} \,|\, \mathbf{w}, b) \;&\propto\; p(\{y_j\} \,|\, \{\vec{\mathbf{d}}_j\}, \mathbf{w}, b) \\ &= \sum_{j:y_j=1} P(M_1 \,|\, \vec{\mathbf{d}}_j) \sum_{j:y_j=2} (1 - P(M_1 \,|\, \vec{\mathbf{d}}_j)) \end{aligned} \tag{5}$$

Although this objective function cannot be optimized in closed-form, it is convex, which means it has a single minimum. Therefore, we can optimize it with some form of gradient descent, for which the initial guess is not critical.

Issues with Class-Conditional and LR Models

Class-Conditional Models:

- The single Gaussian model is often rather crude. PCA coeff's can exhibit significantly more structure (cf. Murase & Nayar).
- A Gaussian model will also be a poor model of non-eye images.
- As a result of this unmodelled structure, detectors based on single Gaussian models are often poor.

Losgistic Regression:

- Discriminative model does note require a model over the observations, and often has fewer parameters as a result.
- LR is an example but its linear decision boundary is only rich enough to limited domains.

Alternatives:

- An alternative approach is to consider warped and aligned view based models (see Cootes, Edwards, & Taylor, 1998).
- Richer density models of the subspace coefficients are possible (e.g., nearest neighbour as in Murase & Nayar, or mixture models).

Breakthrough:

- More sophisticated discriminative models with simple (fast) feature extraction (see Viola & Jones, 2004).

AdaBoost: Binary Classification Problem

Given training data $\{\vec{x}_j, y_j\}_{j=1}^N$, where

- $\vec{x}_j \in \Re^d$ is the feature vector for the j^{th} data item,
- $y_j \in \{-1, 1\}$ denotes the class membership of the j^{th} item $\vec{x}_j$,

we seek a classifier $F(\vec{x})$ such that $y(x) \equiv \text{sign}(F(\vec{x}))$ approximates (in some sense) the training data.

In particular, on the training data, the given class indicator y_j should agree with the model $\text{sign}(F(\vec{x}_j))$ as much as possible.

AdaBoost is an algorithm for greedily training classifiers $F(\vec{x}\,)$ which take the form of additive linear models.

Additive Linear Models

An additive linear model has the form

$$\begin{aligned} F_m(\vec{x}) &= \sum_{k=1}^{m} \alpha_k f_k(\vec{x}; \vec{\theta}_k) \\ &= F_{m-1}(\vec{x}) + \alpha_m f_m(\vec{x}; \vec{\theta}_m). \end{aligned} \tag{6}$$

Here $m \geq 1$ and

- $F_m(\vec{x})$ is a weighted (i.e. α_k) sum of simpler functions $f_k(\vec{x}; \vec{\theta}_k)$.
- Note the simpler functions depend on parameters $\vec{\theta}_k$, which we need to fit along with the weights α_k.
- Here we take the simpler functions $f_k(\vec{x}; \vec{\theta}_k)$ to be weak classifiers, providing values in $\{-1, 1\}$ (e.g., decision stumps).
- We use $F_0(\vec{x}) \equiv 0$ in the recursive definition above.

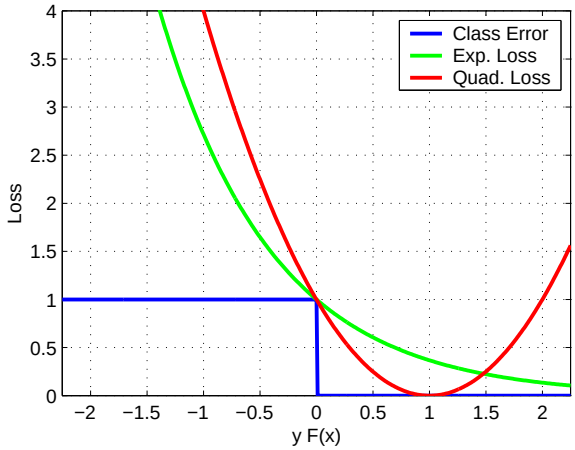
Eigen-appearance models, wavelets, and mixtures of Gaussians models provide simple examples of additive linear models (although, the basis functions are not simply weak classifiers, as they are here).

Exponential Loss

We seek a model $F_m(\vec{x})$ such that $\text{sign}(F_m(\vec{x}_k))$ typically agrees with the class indicator $y_j \in \{-1, 1\}$ in the training data.

How should we measure agreement/disagreement?

Since y_k should have the same sign as $F_m(\vec{x}_k)$, it is convenient to consider $y_k F_m(\vec{x}_k)$, which should be positive.



Possible loss (cost) functions of $z \equiv yF(\vec{x})$ are:

- **Classification Error.** $C(z) = 1$ if $z <= 0$, else 0. Hard to optimize due to discontinuity.
- **Quadratic Loss.** $C(z) = (z-1)^2$. Easy to optimize but penalizes $F(\vec{x})$ for being the correct sign but too large (i.e. confident and correct).
- **Exponential Loss.** $C(z) = exp(-z)$. Smooth and monotonic in z. Large costs for $F(\vec{x})$ large in absolute value, but the wrong sign (i.e. confident and wrong, e.g. data outliers).

Greedy Fitting and AdaBoost

Suppose we have trained a classifier $F_{m-1}(\vec{x})$ with $m-1$ weak components, and wish to add one more linear component,

$$F_m(\vec{x}) \;=\; F_{m-1}(\vec{x}) + \alpha_m f_m(\vec{x};\vec{\theta}_m).$$

Suppose we choose α_m and $\vec{\theta}_m$ to minimize the exponential loss

$$\begin{aligned}\sum_{j=1}^{N} C(y_j F_m(\vec{x}_j)) \;&\equiv\; \sum_{j=1}^{N} e^{-y_j F_m(\vec{x}_j)} \\ &=\; \sum_{j=1}^{N} e^{-y_j F_{m-1}(\vec{x}_j)} e^{-y_j \alpha_m f_m(\vec{x}_j,\vec{\theta}_m)} \\ &=\; \sum_{j=1}^{N} w_j^{m-1} e^{-y_j \alpha_m f_m(\vec{x}_j,\vec{\theta}_m)}\end{aligned}$$

Here the weights $w_j^{m-1} = e^{-y_j F_{m-1}(\vec{x}_j)}$ are just the exponential losses for the previous function $F_{m-1}(\vec{x})$ on each training item.

Note the weights are largest for data points which the previous function $F_{m-1}(\vec{x})$ confidently classifies incorrectly (i.e. $y_j F_{m-1}(\vec{x}_j)$ significantly negative), and are smallest for points confidently classified correctly (i.e. $y_j F_{m-1}(\vec{x}_j)$ significantly positive).

This greedy fitting of the weak classifiers in an additive model leads to the AdaBoost learning algorithm (see Friedman et al, 2000).

AdaBoost Algorithm

for all training exemplars: $j = 1...N$, $w_j^{(1)} = 1$

for $m = 1$ to M **do**

Fit weak classifier m to minimize the objective function:

$$\epsilon_m = \frac{\sum_j w_j^{(m)} I(f_m(\vec{x}_j, \vec{\theta}_m) \neq y_j)}{\sum_j w_j^{(m)}}$$

where $I(f_m(\vec{x}_j) \neq y_j) = 1$ if $f_m(\vec{x}_j) \neq y_j$ and 0 otherwise

$\alpha_m = \ln \frac{1-\epsilon_m}{\epsilon_m}$

for all i **do**

$w_j^{(m+1)} = w_j^{(m)} e^{\alpha_m I(f_m(\vec{x}_j) \neq y_j)}$

end for

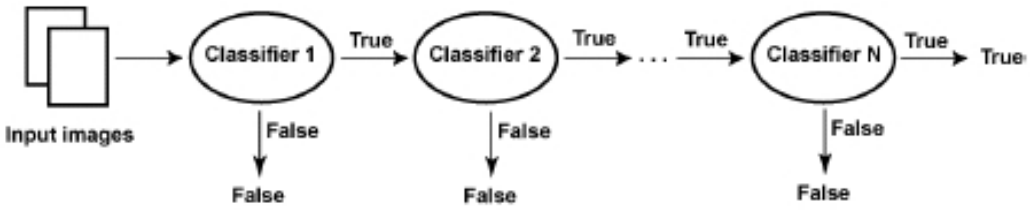
end for

After learning, the final classifier is

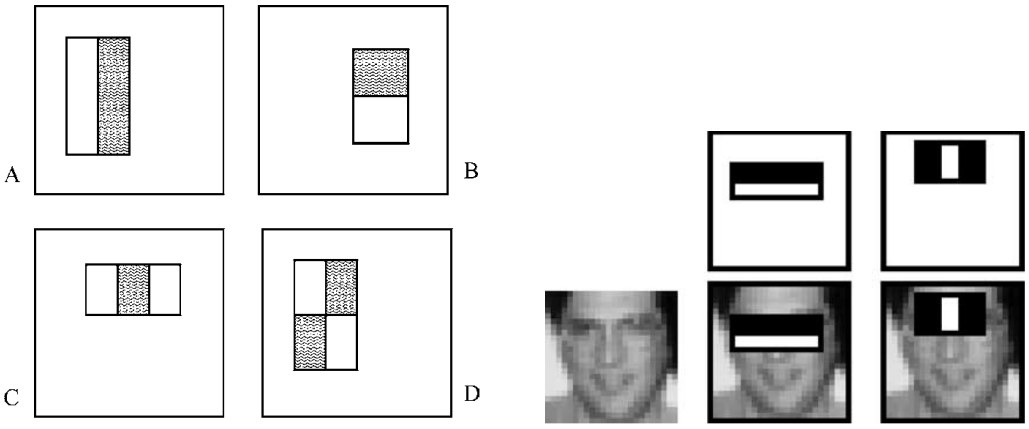
$$g(\vec{x}) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m f_m(\vec{x}, \vec{\theta}_m)\right) \tag{7}$$

Viola and Jones Face Detector

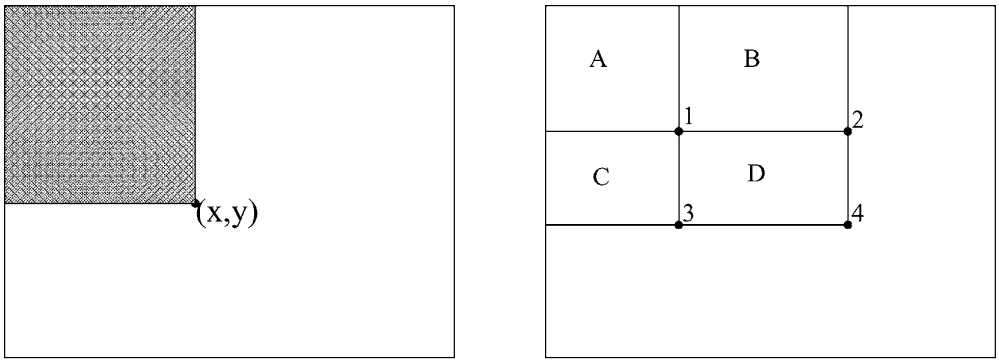
Rejection cascade architecture:



Features are formed from Haar filters...

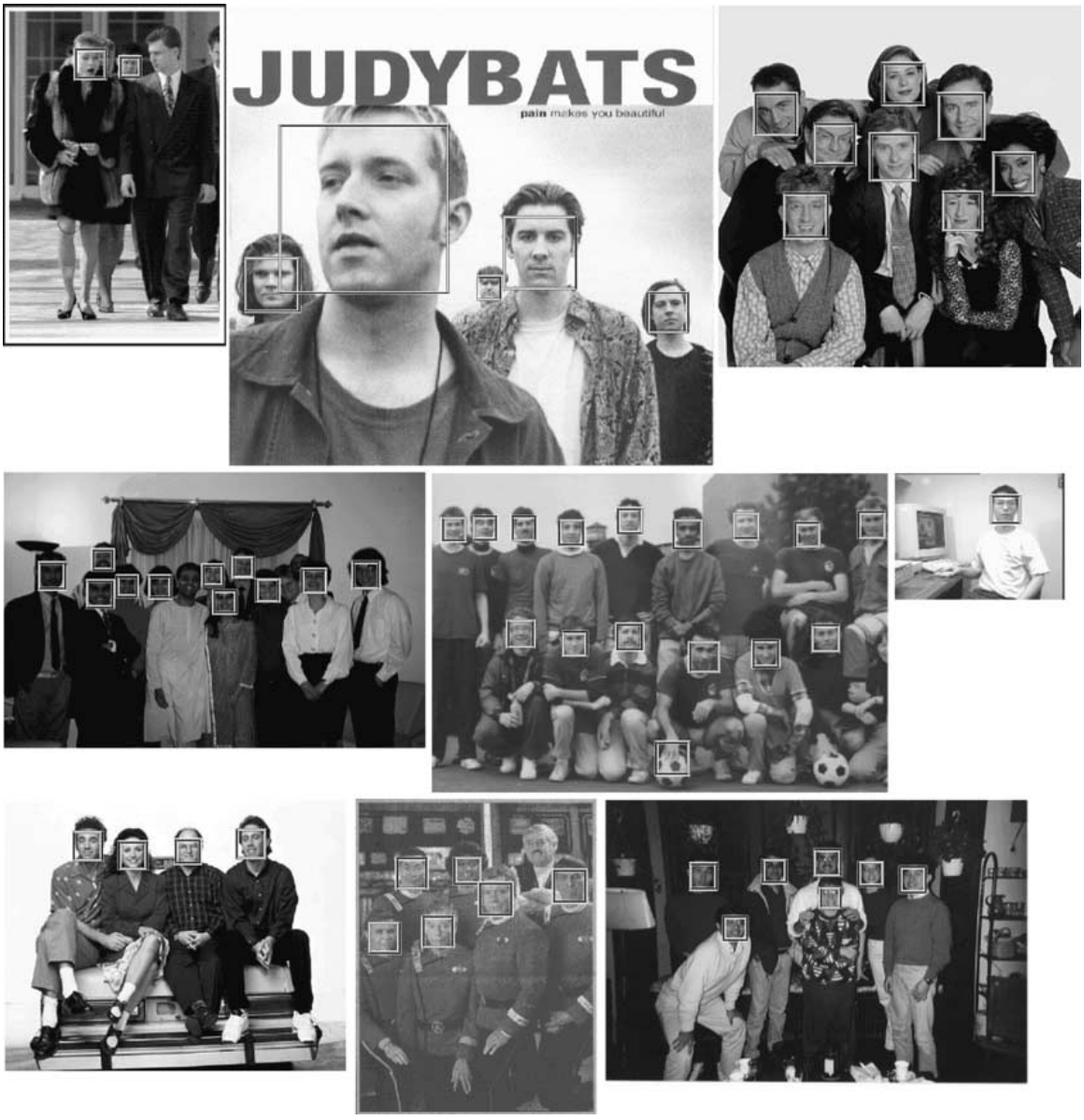


These features can be computed extremely rapidly using integral images.



The result is a real-time face detector with good classification performance (Viola and Jones, 2004).

Viola and Jones, Results



More Results

- boosting for side views
- boosting on HOG features
- boosting on flow features
- boosting shared features

Further Readings

Belhumeur et al (1997) Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Trans. PAMI*, 19(7):711-720

T. Cootes, G. Edwards, and C.J. Taylor, Active Appearance Models, *Proc. IEEE ECCV*, 1998.

J. Friedman, T. Hastie, and R. Tibshirani, Additive logistic regression: a statistical view of boosting, Annals of Statistics 28, 2000, pp. 337-407.

Golub, G. and van Loan, C. (1984) Matrix Computations. Johns Hopkins Press, Baltimore.

T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, Data Mining, Inference, and Prediction, Springer, 2001.

Murase, H. and Nayar, S. (1995) Visual learning and recognition of 3D objects from appearance. *Int. J. Computer Vision* 14:5–24.

Moghaddam, B., Jebara, T. and Pentland, A. (2000) Bayesian face recognition. *Pattern Recognition*, 33(11):1771-1782

Turk, M. and Pendland, A. (1991) Face recognition using eigenfaces, *J. Cognitive Neuroscience*, 3(1):71–86.

P. Viola, and M.J. Jones, Robust real-time face detection, Int. J. of Computer Vision, 2004.